下载APP

10 | 为什么 100% 的测试覆盖率是可以做到的？

2021-08-25 郑晔

《程序员的测试课》 课程介绍 >

讲述：郑晔
时长 10:10 大小 9.32M

你好，我是郑晔！

上一讲我们谈到了测试覆盖率，讲了如何在实际的项目中利用测试覆盖率发现没有覆盖到的代码。最后，我们留下了一个问题：测试覆盖率应该设置成多少？我给出的答案是 100%，但这显然是一个令很多人崩溃的答案。别急，这一讲我们就来说说怎样向着 100% 的测试覆盖率迈进。

很多人对测试覆盖率的反对几乎是本能的，核心原因就是测试覆盖率是一个数字。我在《10x 程序员工作法》中曾经说过，要尽可能地把自己的工作数字化。本来这是一件事，但是，很多管理者就会倾向于把它变成一个 KPI（Key Performance Indicator，关键绩效指标）。KPI 常常是上下级博弈的地方，上级希望高一点，下级希望低一点。所以，从本质上说，很多人对测试覆盖率的反对，首先是源于对 KPI 本能的恐惧。

抛开这种本能的恐惧，我们先来分析一下，如果我们想得到更高质量的代码，测试肯定是越多越好。那多到什么程度算最多呢？答案肯定是 100%。如果把测试覆盖率设置成 100%，就没有那么多扯皮的地方了。比如，你设成了 80%，肯定有人问为啥不设置成 85%；当你设置成 85% 的时候，就会有人问为啥不是 90%，而且他们的理由肯定是一样的：测试覆盖率越高越好。那我设置成 100%，肯定不会有人再问为啥不设置成更高的。

现在你知道了，我们把覆盖率设置成 100% 这应该是极限的标准了。接下来，要回答的一个问题就是，怎么把覆盖率做成 100%。

向 100% 迈进

首先，我们需要明确的一点是，我们用测试覆盖的代码主要是我们自己编写的代码。为什么要强调这一点呢？因为很多时候，我们会涉及使用第三方程序库，而第三方程序库的功能不应该由我们来验证。比如 Jackson 将对象转换为 JSON 是否转得正确，其实我们是不关心的，这是 Jackson 这个程序库要来保证的。

之所以要先强调这一点，因为在很多人编写的代码中，自己编写的业务代码和第三方程序库的代码常常是混杂在一起的。我们工作的重点是，**保证自己编写的代码 100% 测试覆盖。**这意味着什么呢？

首先，让自己可控的代码有完全的测试保证，其次，如果有第三方的代码影响到测试覆盖，我们应该把第三方的代码和我们的代码隔离开。

我知道，很多人已经准备强调 100% 的测试覆盖是如何困难了。其实，不知道你有没有注意，我们在实战环节中，已经完成了一次 100% 的测试覆盖。你可以去看看实战环节的构建脚本，其中用到的测试覆盖率工具就是 JaCoCo，而覆盖率的要求就是 100%，也就是 1.0。问题是我们是怎么做到的呢？

我们不妨一起回想一下，在做好了整体的设计之后，我们每实现一个具体的功能，都考虑了测试的场景，测试用例和代码是同步在实现。最后通过测试覆盖率检查，找出没有覆盖到的代码。对于一些不方便测试的第三方程序库代码，我们进行了隔离，而且要求隔离是非常薄的一层。这样，就保证了我们所有编写业务代码都能够很好地得到测试覆盖。

说起来并不复杂，但你或许会说，这是因为我们只实现了基本的功能，代码复杂度比较低，如果是实现了更为复杂的功能，是不是就没办法覆盖了呢？

我们在前面的内容中说过，要想写好测试，一个关键点是要有良好的软件设计，而且代码本身要尽可能地消除坏味道。到这里你就清楚了，**其实程序员写测试不单单是写测试，同时，也是在发现自己代码中的不足，无论是设计上，还是代码本身。**

所以说，即便是再复杂的功能，通过软件设计和良好的编码，也可以落实到一个一个小代码块上。这里的重点是小，代码能否写短小，这是一个程序员编码基本功的问题。

你让我给一个长达几百上千的代码去写测试，我也很难做到 100% 覆盖，因为代码写得太复杂了，我们理解起来很吃力，为它写测试当然也很吃力。所以，我们会把讨论先集中在一个新项目该如何写测试上。如果一个程序员不能够在干干净净的代码库上写好代码，你就很难指望他在面对一个遗留代码库时能够写好代码。

不知道你注意到了没有，我们说在实战中达成 100% 测试覆盖时，还有一个工作习惯，就是测试和代码同步写。为什么要这么做呢？因为没有人愿意补测试，无论这个代码是你写的还是别人写的。

这也就是为什么要把测试放在自动化过程中，这样，我们每完成一个任务，就要确保编写了相应的测试。而且，我前面也强调过，任务的关键是小，比如，小到半个小时就可以提交一次，这样，你写测试的负担相对来说是小的。小事相比大事更容易坚持，这是符合人性的做法。

你现在已经知道了，一个新项目想要达到 100% 的测试覆盖，**首先，要有可测试的设计，要能够编写整洁的代码；其次，测试和代码同步写。**

测不到的代码

关于 100% 测试覆盖率，很多人有一个误区：100% 覆盖了，是不是就意味着代码没问题了？答案是否定的。即便我们有了 100% 的测试覆盖，还是会有你想不到的场景出现。100% 的覆盖只是保证我们已经写的代码没有场景遗漏，不会有异常场景没有处理，不会有分支条件没有考虑到，仅此而已。

100% 的测试覆盖只是程序员做好了本职工作，保证了在这个环节内没有出错。而软件整体质量是一个系统性的工程，首先要保证我们尽可能多地考虑到了各种测试场景，这是我们在第 3 讲中讨论的内容。

对程序员来说，通过把测试覆盖率设置 100%，我们就有了一个查缺补漏的机会。一旦发现有些缺漏很难补上怎么办？就像我们在实战环节中见到的那样，模拟 Jackson 的异常成本过高，我们就会采用隔离的方式，将不好测试的地方隔离开来，形成一个封装层。实际上，我们是在用软件设计的方式在解决问题。

理解了达成 100% 测试覆盖的基础之后，我还必须再强调一下。第一点是前面提到的封装层，这一层一定要非常薄。很多情况下，可能就是直接的方法调用。如果有复杂的逻辑，比如在防腐层代码中有对象之间的转换，我们都可以把转换的逻辑拿出来，单独地去写测试，因为这个转换逻辑多半是可以测试的。100% 的测试覆盖率我们不是说说而已，而是要坚持做到能覆盖的尽量去覆盖。

另外还有一点，隔离出来的代码怎么办呢？我们要在测试覆盖的检查中将它们排除，具体的做法就是在构建文件中，把这个文件标记为不需要测试覆盖。

复制代码

```
coverage {
  excludeClasses = [
        "com.github.dreamhead.todo.util.Jsons"
  ]
}
```

在我的项目中，我会要求这里只能有那个薄薄的封装层。有些初次接触项目的人，常常会把这里理解成项目中有我不想测的代码，却还要保证 100% 测试覆盖，这里就是一种妥协。绝对不是这个意思！所以，一方面，我们要在团队中强调这个纪律，另一方面，我们也要经常性地做代码评审，保证这个用来隔离封装层的地方不会遭到滥用。

100% 虽然要求很高，但要想做到，首先是理念上的认同，然后，我们就可以想各种办法去做到。在实际的项目中，**很多人先从理念去否定，认为不可能做到，只要有一点困难就放弃，这其实才是 100% 测试覆盖率难以达成的最主要原因。**

总结时刻

今天我们延续了上一讲测试覆盖率的话题，讨论了在一个新项目中，测试覆盖率应该设置成多少，我给出的答案就是 100%。

100% 的测试覆盖率会遭到很多人的反对，但这种反对首先是对 KPI 行为的一种本能恐惧。在真实项目中，大家都认同的观点是测试覆盖率越高越好，最高的覆盖率肯定是 100%。

我们强调的 100% 测试覆盖，主要指的是对自己编写的代码 100% 测试覆盖。这就意味着，我们一方面要保证自己的代码完全可控，另一方面，对于影响到测试覆盖的第三方代码要进行隔离。要想做到 100% 的测试覆盖，技术上说，要有可测试的设计以及编写整洁的代码，实践上看，要测试和代码同步产出。

100% 的测试覆盖并不是说代码没有问题了，而应该是程序员对自己编写代码的一种质量保证，它是一个帮助我们查缺补漏的过程。

对于无法测试到第三方代码，要用一个薄薄的隔离层将代码隔离出去，在构建脚本中将隔离层排除在外。有一点需要注意的是，排除脚本千万别被滥用了。

如果今天的内容你只能记住一件事，那请记住：**100% 的测试覆盖率是程序员编写高质量代码的保证。**

思考题

今天我们讲了如何达到 100% 的测试覆盖，你在实际工作中遇到过哪些难以测试的情况呢？期待在留言区看到你的想法。

分享给需要的人，Ta订阅后你可得 20 元现金奖励

赞 2 提建议

© 版权归极客邦科技所有，未经许可不得传播售卖。 页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 09 | 测试覆盖率：如何找出没有测试到的代码？

下一篇 11 | 集成测试：单元测试可以解决所有问题吗？

精选留言 (6)

写留言

coyang
2021-08-26

老师好，我遇到以下两种不好测试的情况：
1.死循环的逻辑
2.static函数

以上两种情况怎么保证覆盖率100%呢？

展开

作者回复: 死循环，把逻辑部分挪出来，单独测试。把循环的地方单独隔离出来，排除在覆盖率之外。

static 函数，要是没有任何字段，只是纯逻辑，就可以直接测试，就像一个程序库去测试。要是有复杂的关联，还是最好先考虑改造吧，把依赖解开，改造成一个正常的函数再做测试。这类似于遗留系统，后面我会提到，遗留系统改造主要是分离和隔离。

1 1

2021-08-26

TDD写出来的代码一般情况覆盖率都是100%

展开

作者回复: TDD 比较容易 100%

davix
2021-08-26

請問mock中用verify 的多少影響覆蓋率嗎

展开

作者回复: 不会

davix
2021-08-25

請問100%覆蓋會不會導致過度測試導致難以重構？

展开

作者回复: 往前看，这也是我为什么要先讲做好设计，做好单元测试的原因，有了这些基础，才是100%覆盖。

1

Alexdown
2021-08-25

“我前面也强调过，任务的关键是小，比如，小到半个小时就可以提交一次，这样，你写测试的负担相对来说是小的”，请问这里的“提交”指的是git commit(提交到本地库)还是git push（提交到远程库）。

我个人的理解是git commit，完成某个功能的最后一个子任务后，将该功能的多个git c...

展开

作者回复: 我没有做区分，能 commit 和能 push 对我来说是一回事，就是一次提交，一个完整任务的完成。push 才是终点，重点是把自己的代码是和别人的代码合在一起，否则，合并的工作量会增大。

1

大碗
2021-08-25

请问老师对“第三方代码”的界定为是不是自己编写的代码，第一种常见的是非公司的类库，比如Jackson，这种好理解。第二种是服务A调用服务B的api.jar，第三种两个人编写服务A的两个模块C，D，编写C的人调用了别人D的方法，后面两种也属于调用第三方代码么?

展开

作者回复: 我指的就是你没法控制的代码。